# Task 6.3DN Sorting Algorithms — Timing Report

**Name:** Thomas Couser

**ID:** 692529

## Experimental Conditions

**Operating System:** Windows

**Processor:** i7-8700K

**Python version:** 3.12.2

## Unsorted Data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algorithm** | **Trial time (ms)** | | | | | **Median** |
| **1** | **2** | **3** | **4** | **5** |
| Bubble | 127592 | 128118 | 130445 | 130939 | 126204 | 127592 |
| Selection | 55679 | 56775 | 58858 | 55997 | 56052 | 56052 |
| Insertion | 49095 | 48505 | 50745 | 48400 | 47207 | 48505 |
| List.sort (Powersort) | 5.6 | 5.7 | 5.6 | 5.5 | 5.5 | 5.6 |

## Already Sorted Data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algorithm** | **Trial time (ms)** | | | | | **Median** |
| **1** | **2** | **3** | **4** | **5** |
| Bubble | 84566 | 83097 | 82908 | 77478 | 75157 | 82908 |
| Selection | 59194 | 60191 | 59446 | 59073 | 54444 | 59194 |
| Insertion | 830 | 826 | 837 | 819 | 781 | 826 |
| List.sort (Powersort) | 0.2 | 0.3 | 0.2 | 0.3 | 0.4 | 0.3 |

## Discussion

Bubble sort is by far the least efficient sorting algorithm of the examples. Python built in sort algorithm is by far the most efficient. The biggest difference when the data is sorted or unsorted can be seen with insertion sort and power sort, with a difference factor of 59 and 19 respectively. Selection sort performed about the same, while bubble sort took a bit longer for unsorted data. The conclusion is to not use bubble sort.